Implement K-Nearest Neighbors algorithm on diabetes.csv dataset. Compute confusion matrix, accuracy, error rate, precision and recall on the given dataset

In [1]:
```
import numpy as np
import pandas as pd
```

In [2]:
```
data = pd.read_csv('7458_diabetes.csv')
data.head()
```

Out[2]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	Pedigree	Age	Outc
0	6	148	72	35	0	33.6	0.627	50	
1	1	85	66	29	0	26.6	0.351	31	
2	8	183	64	0	0	23.3	0.672	32	
3	1	89	66	23	94	28.1	0.167	21	
4	0	137	40	35	168	43.1	2.288	33	

In [3]:
```
#Check for null or missing values
data.isnull().sum()
```

Out[3]:
```
Pregnancies      0
Glucose          0
BloodPressure    0
SkinThickness    0
Insulin          0
BMI              0
Pedigree         0
Age              0
Outcome          0
dtype: int64
```

In [4]:
```
# Select only numeric columns to avoid errors with .mean()
numeric_cols = data.select_dtypes(include=np.number).columns

for column in numeric_cols:
    # Use direct assignment to replace 0s
    data[column] = data[column].replace(0, np.nan)

    # Calculate the mean for the column
    mean_value = round(data[column].mean(skipna=True))

    # Use direct assignment to fill NaN values
    data[column] = data[column].fillna(mean_value)
```

```
data.head(10)
```

Out[4]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	Pedigree	Age	Outc
0	6.0	148.0	72.0	35.0	156.0	33.6	0.627	50	
1	1.0	85.0	66.0	29.0	156.0	26.6	0.351	31	
2	8.0	183.0	64.0	29.0	156.0	23.3	0.672	32	
3	1.0	89.0	66.0	23.0	94.0	28.1	0.167	21	
4	4.0	137.0	40.0	35.0	168.0	43.1	2.288	33	
5	5.0	116.0	74.0	29.0	156.0	25.6	0.201	30	
6	3.0	78.0	50.0	32.0	88.0	31.0	0.248	26	
7	10.0	115.0	72.0	29.0	156.0	35.3	0.134	29	
8	2.0	197.0	70.0	45.0	543.0	30.5	0.158	53	
9	8.0	125.0	96.0	29.0	156.0	32.0	0.232	54	

In [5]:
```
X = data.iloc[:, :8] #Features
Y = data.iloc[:, 8:] #Predictor
```

In [6]:
```
from sklearn.model_selection import train_test_split

# X is your full feature DataFrame, Y is your full target Series
X_train, X_test, Y_train, Y_test = train_test_split(
    X, Y,
    test_size=0.2,    # Or whatever your test size is
    random_state=42,  # Good practice for reproducible results
    stratify=Y        # <-- THIS IS THE FIX
)
```

In [7]:
```
#KNN
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier()
knn_fit = knn.fit(X_train, Y_train.values.ravel())
knn_pred = knn_fit.predict(X_test)
```

In [8]:
```
from sklearn.metrics import confusion_matrix, precision_score, recall_score, f1_sco
print("Confusion Matrix")
print(confusion_matrix(Y_test, knn_pred))
print("Accuracy Score:", accuracy_score(Y_test, knn_pred))
print("Reacal Score:", recall_score(Y_test, knn_pred))
print("F1 Score:", f1_score(Y_test, knn_pred))
print("Precision Score:",precision_score(Y_test, knn_pred))
```

```
Confusion Matrix
[[154]]
Accuracy Score: 1.0
Reacal Score: 1.0
F1 Score: 1.0
Precision Score: 1.0
```

```
C:\Users\swast\AppData\Local\Programs\Python\Python313\Lib\site-packages\sklearn\metrics\_classification.py:407: UserWarning: A single label was found in 'y_true' and 'y_pred'. For the confusion matrix to have the correct shape, use the 'labels' parameter to pass all known labels.
  warnings.warn(
```

In [ ]:

In [ ]: